```
* DES example, osteoporosis model, NICE DSU TSD15 Patient Level Simulation ;
* with deterministic and probabilistic sensitivity analysis (DSA);
* 2018.11.20;

* Initial settings;
data _null_; * for basecase analysis;

  * DES settings;
    des_seed = 2370;         * seed for DES, only the first record of this dataset is effective in simulation;
    nPts     = 1000000;      * number of patients simulated in DES, same patient will be evaluated both control and intervention
cohorts;
    outIdv   = 1;            * if =1 (^=0), output individual data. =0, do not;
    nInner   = 0;            * number of INNER loop to check the tte variables;
    maxWTP   = 100000;      * maximum WTP for CEAC;

    call symput("des_seed", trim(left(put(des_seed, 24.))));
    call symput("nPts"    , trim(left(put(nPts    ,  8.))));
    call symput("outIdv"  , trim(left(put(outIdv  ,  8.))));

run;

data _null_; * for DSA;

  * DES settings;
    des_seed = 4358;         * seed for DES, only the first record of this dataset is effective in simulation;
    nPts     = 1000000;     * number of patients simulated in DES, same patient will be evaluated both control and intervention
cohorts;
    outIdv   = 0;            * if =1 (^=0), output individual data. =0, do not;
    nInner   = 0;            * number of INNER loop to check the tte variables;
    maxWTP   = 100000;      * maximum WTP for CEAC;

    call symput("des_seed", trim(left(put(des_seed, 24.))));
    call symput("nPts"    , trim(left(put(nPts    ,  8.))));
    call symput("outIdv"  , trim(left(put(outIdv  ,  8.))));

run;

data parms; * for basecase, DSA;

    type   = "Base    ";

    cHfrac = 7000; * cost for a hip fracture event in GBP;
    cVfrac = 3000; * cost for a vertebral fracture event in GBP;
    cDrug  = 1000; * annual cost of drug intervention in GBP;
                   * originally 500 per annum that results in cost-saving with better QALYs, ie, dominant;
    uBL    = .7;   * baseline utility;
    disuH  = .75;  * utility multiplier for hip fracture;
```

```
    disuV1 = .9;   * utility multiplier for 1st vertebral fracture;
    disuV2 = 1;    * utility multiplier for 2nd vertebral fracture; * no decrease;
    wshape = 1.25; * parameter for time form baseline to hip fracture, shape parameter (alpha) of Weibull distribution;
                   * assume identical for hip and vertebral, originally 4 and 2, respectively;
    wscaleH = 10;   * parameter for time form baseline to hip fracture, location parameter (beta) of Weibull distribution;
    wscaleV = 8;    * parameter for time form baseline to vertebral fracture, location parameter of Weibull distribution;
    wshapeM = 1.3;
    wscaleM = 16;
    pDthH  = .05;  * probability for death due to hip fracture;
    rrH    = .5;   * treatment effect, accelerated factor of intervention for hip fracture;
    rrV1   = .5;   * treatment effect, accelerated factor of intervention for 1st vertebral fracture;
    rrV2   = 1;    * treatment effect, accelerated factor of intervention for 2nd vertebral fracture;
    hrzn   = 30;   * time horizon (years);
    Disc   = .035;  * annual discount rate;

    des_seed = &des_seed;
    nPts     = &nPts;
    outIdv   = &outIdv;

run;

data parms; * for DSA;
  set parms;
  output;
  type = "DSA01"; _value = cHfrac ; cHfrac =  4000; output; cHfrac  = _value;
  type = "DSA02"; _value = cHfrac ; cHfrac = 10000; output; cHfrac  = _value;
  type = "DSA03"; _value = cVfrac ; cVfrac =  2000; output; cVfrac  = _value;
  type = "DSA04"; _value = cVfrac ; cVfrac =  5000; output; cVfrac  = _value;
  type = "DSA05"; _value = uBL    ; uBL    = .6   ; output; uBL     = _value;
  type = "DSA06"; _value = uBL    ; uBL    = .8   ; output; uBL     = _value;
  type = "DSA07"; _value = disuH  ; disuH  = .65  ; output; disuH   = _value;
  type = "DSA08"; _value = disuH  ; disuH  = .85  ; output; disuH   = _value;
  type = "DSA09"; _value = disuV1 ; disuV1 = .8   ; output; disuV1  = _value;
  type = "DSA10"; _value = disuV1 ; disuV1 = .95  ; output; disuV1  = _value;
  type = "DSA11"; _value = wshape ; wshape = 1    ; output; wshape  = _value;
  type = "DSA12"; _value = wshape ; wshape = 3    ; output; wshape  = _value;
  type = "DSA13"; _value = wshapeM; wshapeM = 1   ; output; wshapeM = _value;
  type = "DSA14"; _value = wshapeM; wshapeM = 1.5 ; output; wshapeM = _value;
  type = "DSA15"; _value = wscaleM; wscaleM = 12  ; output; wscaleM = _value;
  type = "DSA16"; _value = wscaleM; wscaleM = 20  ; output; wscaleM = _value;
  type = "DSA17"; _value = pDthH  ; pDthH  = .01  ; output; pDthH   = _value;
  type = "DSA18"; _value = pDthH  ; pDthH  = .1   ; output; pDthH   = _value;
  type = "DSA19"; _value = rrH    ; rrH    = .30  ; output; rrH     = _value;
  type = "DSA20"; _value = rrH    ; rrH    = .75  ; output; rrH     = _value;
  type = "DSA21"; _value = rrV1   ; rrV1   = .30  ; output; rrV1    = _value;
  type = "DSA22"; _value = rrV1   ; rrV1   = .75  ; output; rrV1    = _value;
  drop _:;
```

```
run;


data individual(keep = type i j tt_: death_Hfrac Hfrac Vfrac Death LY QALY dLY dQALY Cost dCost CostF dCostF CostI dCostI actual_tt:)
/* for basecase, DSA*/
      summary   (keep = type nPts mean: motality: inc: ICERqaly dICERqaly Quadrant dQuadrant);

  call streaminit( &des_seed ); * seed for random number generator;

  set parms;

  iDisc = log(1 + Disc);  * instantaneous equivalent of the annual discount rate, see NICE DSU TSD15;

  array cInt{2}; cInt{1} = 0; cInt{2} = cDrug; * cost for intervention;
  array tte{2, 5};      * random variable (1 to 4 for time to death, hip, 1st and 2nd vert frac, and 5 for death due to hip frac);
  array mean_LY{2};     * mean life years without discounting;
  array meanDLY{2};     * mean life years with    discounting;
  array mean_QALY{2};   * mean QALY without discounting;
  array meanDQALY{2};   * mean QALY with   discounting;
  array mean_Cost{2};   * mean Cost without discounting;
  array meanDCost{2};   * mean Cost with    discounting;
  array mean_CostF{2};  * mean Cost for fractures without discounting;
  array meanDCostF{2};  * mean Cost for fractures with    discounting;
  array mean_CostI{2};  * mean Cost for interventions without discounting;
  array meanDCostI{2};  * mean Cost for interventions with    discounting;
  array meanHfrac{2};   * mean number of hip fractures;
  array meanVfrac{2};   * mean number of vertebral fractures;
  array Motality{2};    * percentage of dead during time horizon;

  i = .;         * i for patients pair;
  do j = 1 to 2; * j for interventions, 1 control, 2 drug intervention;
    mean_LY{j} = 0; meanDLY{j} = 0; mean_QALY{j} = 0; meanDQALY{j} = 0;
    mean_Cost{j} = 0; meanDCost{j} = 0; mean_CostF{j} = 0; meanDCostF{j} = 0; mean_CostI{j} = 0; meanDCostI{j} = 0;
    meanHfrac{j} = 0; meanVfrac{j} = 0; Motality{j} = 0;
  end;

  do i = 1 to nPts; * loop for a pair of patients for control and intervention with same random numbers for time-to-events;

    * generating patient specific time-to-event random numbers, and applying treatment effects of drug intervention;
    * control cohort;
    tte{1, 1} = rand("weibull", wshapeM, wscaleM); * time to death;
    tte{1, 2} = rand("weibull", wshape,  wscaleH); * time to hip fracture;
    tte{1, 3} = rand("weibull", wshape,  wscaleV); * time to 1st vertebral fracture;
    tte{1, 4} = rand("weibull", wshape,  wscaleV); * time to 2nd vertebral fracture from 1st vertebral facture;
    tte{1, 5} = rand("binomial", pDthH, 1);      * death due to hip fracture;

    * drug intervention cohort;
```

```
    tte{2, 1} = tte{1, 1};
    tte{2, 2} = tte{1, 2} / rrH;
    tte{2, 3} = tte{1, 3} / rrV1;
    tte{2, 4} = tte{1, 4} / rrV2 + tte{2, 3};      * time from baseline to 2nd vertebral fracture;
    tte{2, 5} = tte{1, 5};

    * control cohort;
    tte{1, 4} = tte{1, 3} + tte{1, 4};             * time from baseline to 2nd vertebral fracture;

    do j = 1 to 2; * loop for intervention, examining a pair of patients for control and drug intervention;

      Utility = uBL;
      LY = 0; dLY = 0; QALY = 0; dQALY = 0; Cost = 0; dCost = 0; CostF = 0; dCostF = 0; CostI = 0; dCostI = 0; Death = 0; Hfrac = 0;
Vfrac = 0;

      * time of the 1st event;
      current = 0;                                                  * time variable indicates PREVIOUS event;
      next    = min(tte{j, 1}, tte{j, 2}, tte{j, 3}, tte{j, 4}); * time variable indicates NEXT event;

      do until (Death = 1 or next >= hrzn); * loop until death or reach time horizon, examining event history of a patient;

        if next = tte{j, 1} then do; * death;
          Death = 1;
           LY   =  LY   +           (next - current);
           QALY =  QALY + Utility * (next - current);
          dLY   = dLY   +       1 / (-iDisc) * (exp(-iDisc * next) - exp(-iDisc * current));  * continuous discount, see NICE DSU
TSD15;
          dQALY = dQALY + Utility / (-iDisc) * (exp(-iDisc * next) - exp(-iDisc * current));
          tte{j, 1} = tte{j, 1} + 10000; * mark event happened;
        end;

        else if next = tte{j, 2} then do; * hip fracture;
          Hfrac = 1;
          if tte{j, 5} = 1 then Death = 1; * hip fracture results in death;
           LY    =  LY    +           (next - current);
           QALY  =  QALY  + Utility * (next - current);
          dLY    = dLY    +       1 / (-iDisc) * (exp(-iDisc * next) - exp(-iDisc * current));
          dQALY  = dQALY  + Utility / (-iDisc) * (exp(-iDisc * next) - exp(-iDisc * current));
           CostF =  CostF + cHfrac;
          dCostF = dCostF + cHfrac / (1 + Disc) ** next;

          Utility = Utility * disuH;
          tte{j, 2} = tte{j, 2} + 10000; * mark event happened;
        end;

        else if next = tte{j, 3} then do; * 1st vert fracture;
          Vfrac = Vfrac + 1;
```

```
     LY    =  LY    +           (next - current);
     QALY  =  QALY  + Utility * (next - current);
    dLY    = dLY    +       1 / (-iDisc) * (exp(-iDisc * next) - exp(-iDisc * current));
    dQALY  = dQALY  + Utility / (-iDisc) * (exp(-iDisc * next) - exp(-iDisc * current));
     CostF =  CostF + cVfrac;
    dCostF = dCostF + cVfrac / (1 + Disc) ** next;

    Utility = Utility * disuV1;
    tte{j, 3} = tte{j, 3} + 10000; * mark event happened;
  end;

  else if next = tte{j, 4} then do; * 2nd vert fracture;
    Vfrac = Vfrac + 1;
     LY    =  LY   +           (next - current);
     QALY  =  QALY + Utility * (next - current);
    dLY    = dLY   +       1 / (-iDisc) * (exp(-iDisc * next) - exp(-iDisc * current));
    dQALY  = dQALY + Utility / (-iDisc) * (exp(-iDisc * next) - exp(-iDisc * current));
     CostF =  CostF + cVfrac;
    dCostF = dCostF + cVfrac / (1 + Disc) ** next;

    Utility = Utility * disuV2;
    tte{j, 4} = tte{j, 4} + 10000; * mark event happened;
  end;

  * next event;
  current = next;
  next    = min(tte{j, 1}, tte{j, 2}, tte{j, 3}, tte{j, 4});

end; * this patient loop;

if Death = 0 then do; * survive until the end of time horizon;
    next = hrzn;
     LY    =  LY   +           (next - current);
     QALY  =  QALY + Utility * (next - current);
    dLY    = dLY   +       1 / (-iDisc) * (exp(-iDisc * next) - exp(-iDisc * current));
    dQALY  = dQALY + Utility / (-iDisc) * (exp(-iDisc * next) - exp(-iDisc * current));
end;

do k = 1 to 4; if tte{j, k} >= 10000 then tte{j, k} = tte{j, k} - 10000; end; * restore to original value;

* cost for intervention (constant overtime until death);
 CostI = cInt{j} * LY;
dCostI = cInt{j} / (-iDisc) * (exp(-iDisc * current) - exp(-iDisc * 0));
 Cost  =  CostF +  CostI;
dCost  = dCostF + dCostI;

* accumulate costs, LYs, QALYs, number of events;
```

```
    mean_CostF{j} = mean_CostF{j} +  CostF / nPts;
    meanDCostF{j} = meanDCostF{j} + dCostF / nPts;
    mean_CostI{j} = mean_CostI{j} +  CostI / nPts;
    meanDCostI{j} = meanDCostI{j} + dCostI / nPts;
    mean_LY{j}    = mean_LY{j}    +  LY    / nPts;
    meanDLY{j}    = meanDLY{j}    + dLY    / nPts;
    mean_QALY{j}  = mean_QALY{j}  +  QALY  / nPts;
    meanDQALY{j}  = meanDQALY{j}  + dQALY  / nPts;

    meanHfrac{j}  = meanHfrac{j}  + Hfrac  / nPts;
    meanVfrac{j}  = meanVfrac{j}  + Vfrac  / nPts;
    Motality{j}   = Motality{j}   + Death  / nPts;

    * output individual patient data;
    if outIdv not= 0 then do;

      * individual time to event data simulated;
      tt_death    = tte{j, 1};
      tt_Hfrac    = tte{j, 2};
      tt_Vfrac1   = tte{j, 3};
      tt_Vfrac2   = tte{j, 4};
      death_Hfrac = tte{j, 5};

      * time of event actually happened;
           if Death not   = 1         then actual_tt_death  = hrzn; * tempolary set time horizon;
      else if death_Hfrac = 0         then actual_tt_death  = tt_death;
      else                                 actual_tt_death  = min(tt_death, tt_Hfrac);
      if tt_Hfrac  <= actual_tt_death then actual_tt_Hfrac  = tt_Hfrac ; else actual_tt_Hfrac  = .;
      if tt_Vfrac1 <= actual_tt_death then actual_tt_Vfrac1 = tt_Vfrac1; else actual_tt_Vfrac1 = .;
      if tt_Vfrac2 <= actual_tt_death then actual_tt_Vfrac2 = tt_Vfrac2; else actual_tt_Vfrac2 = .;
      if Death not = 1                then actual_tt_death  = .;   * alive at the end of time horizon;

      output individual;

    end;

  end; * Tx loop;

end; * patient repeats;

do j = 1 to 2; * sum cost for fractures and intervention;
  mean_Cost{j} = mean_CostF{j} + mean_CostI{j};
  meanDCost{j} = meanDcostF{j} + meanDCostI{j};
end;

* incremental QALYs and costs;
inc_QALY = mean_QALY{2} - mean_QALY{1};
```

```
  inc_Cost = mean_Cost{2} - mean_Cost{1};
  incDQALY = meanDQALY{2} - meandQALY{1};
  incDCost = meanDCost{2} - meanDCost{1};

  * ICER as cost per QALYs;
   Q =   1 * (inc_Cost >= 0 ) * (inc_QALY >= 0)
       + 2 * (inc_Cost >= 0 ) * (inc_QALY <  0)
       + 3 * (inc_Cost <  0 ) * (inc_QALY <  0)
       + 4 * (inc_Cost <  0 ) * (inc_QALY >= 0);
  dQ =   1 * (incDCost >= 0 ) * (incDQALY >= 0)
       + 2 * (incDCost >= 0 ) * (incDQALY <  0)
       + 3 * (incDCost <  0 ) * (incDQALY <  0)
       + 4 * (incDCost <  0 ) * (incDQALY >= 0);

  if  Q = 1 then  ICERqaly = (mean_Cost{2} - mean_Cost{1}) / (mean_QALY{2} - mean_QALY{1}); else  ICERqaly = .;
  if dQ = 1 then dICERqaly = (meanDCost{2} - meanDCost{1}) / (meanDQALY{2} - meanDQALY{1}); else dICERqaly = .;

  select ( Q);
    when (1)   Quadrant = "1                  ";
    when (2)   Quadrant = "2 (dominated)      ";
    when (3)   Quadrant = "3 (low cost & eff.)";
    when (4)   Quadrant = "4 (DOMINANT)       ";
    otherwise  Quadrant = ". (something wrong)";
  end;
  select (dQ);
    when (1)  dQuadrant = "1                  ";
    when (2)  dQuadrant = "2 (dominated)      ";
    when (3)  dQuadrant = "3 (low cost & eff.)";
    when (4)  dQuadrant = "4 (DOMINANT)       ";
    otherwise dQuadrant = ". (something wrong)";
  end;

  output summary;

run;


* for Base;
title1 "DES example: model parameters";
proc transpose data=parms out=parms2; run;
proc print data=parms2; run;
title1 "DES example: summary results";
proc transpose data=summary(obs=30) out=summary2; run;
proc print data=summary2; run;


* IF outIdv = 1;
```

```
title1 "DES example: simulated individual patient, first 10 patients";
proc print data=individual(obs=20); run;

data individual2; set individual(obs=40); if actual_tt_death = . then actual_tt_death = 100; if tt_death > 30 then tt_death = .; run;
proc sql; create index j on individual2; quit;
title1 "DES example: simulated individual patient, first 20 patients";
proc sgpanel data=individual2;
    panelby j / columns = 2; label j = "Treatment strategy";
    hbarparm category = i response = actual_tt_death / barwidth=.5 legendlabel = "Survival";
    rowaxis type = discrete grid label = "Patient Number";
    colaxis type = linear   grid label = "Years" values=(0 to 30 by 5);
    scatter X =        tt_Hfrac  Y = i / legendlabel = "Hip fracture (unhappend)"         markerattrs = (symbol = triangle       size
= 9 color = red);
    scatter X = actual_tt_Hfrac  Y = i / legendlabel = "Hip fracture"                     markerattrs = (symbol = trianglefilled size
= 9 color = red);
    scatter X =        tt_Vfrac1 Y = i / legendlabel = "Vertebral fracture 1 (unhappend)" markerattrs = (symbol = triangle       size
= 9 color = blue);
    scatter X = actual_tt_Vfrac1 Y = i / legendlabel = "Vertebral fracture 2"             markerattrs = (symbol = trianglefilled size
= 7 color = blue);
    scatter X =        tt_Vfrac2 Y = i / legendlabel = "Vertebral fracture 2 (unhappend)" markerattrs = (symbol = triangle       size
= 9 color = green);
    scatter X = actual_tt_Vfrac2 Y = i / legendlabel = "Vertebral fracture 1"             markerattrs = (symbol = trianglefilled size
= 7 color = green);
    scatter X =        tt_death  Y = i / legendlabel = "Death (unhappend)"               markerattrs = (symbol = circle         size
= 9 color = black);
    scatter X = actual_tt_death  Y = i / legendlabel = "Death"                           markerattrs = (symbol = circlefilled   size
= 7 color = black);
run;

* for DSA;
data tornado;
  merge parms summary(keep = type ICERqaly dICERqaly);
  length parameter $8.;
  array  ICER{2};
  array dICER{2};
  retain i ICER: dICER:;
  if _n_ = 1 then do;
    i = 1;
    call symput("base_ICER", trim(left(put( ICERqaly, 12.2))));
    call symput("baseDICER", trim(left(put(dICERqaly, 12.2))));
    output;
  end;
  else do;
         if type in ("DSA01", "DSA02") then parameter = "cHfrac";
    else if type in ("DSA03", "DSA04") then parameter = "cVfrac";
    else if type in ("DSA05", "DSA06") then parameter = "uBL";
    else if type in ("DSA07", "DSA08") then parameter = "disuH";
```

```
    else if type in ("DSA09", "DSA10") then parameter = "disuV1";
    else if type in ("DSA11", "DSA12") then parameter = "wshape";
    else if type in ("DSA13", "DSA14") then parameter = "wshapeM";
    else if type in ("DSA15", "DSA16") then parameter = "wscaleM";
    else if type in ("DSA17", "DSA18") then parameter = "pDthH";
    else if type in ("DSA19", "DSA20") then parameter = "rrH";
    else if type in ("DSA21", "DSA22") then parameter = "rrV1";
     ICER(i) =  ICERqaly;
    dICER(i) = dICERqaly;
    if i = 1 then i = 2;
    else do;
       Length = abs( ICER{1} -  ICER{2});
      dLength = abs(dICER{1} - dICER{2});
      output;
      i = 1;
    end;
  end;
run;
proc sort data=tornado out= tornado; by descending  Length; run;
proc sort data=tornado out=dtornado; by descending dLength; run;

title1 "DES example: DSA model parameters";
proc transpose data=parms out=parms2; run;
proc print data=parms2; run;

title1 "DES example: DSA summary results";
proc transpose data=summary(obs=30) out=summary2; run;
proc print data=summary2; run;

title1 "DES example: DSA Tornado diagram";
proc sgplot data = tornado;
   label ICER1 = "ICER with LOWER parameter value" ICER2 = "ICER with HIGHER parameter value";
   hbar parameter / response = ICER1 datalabel baseline = &base_ICER;
   hbar parameter / response = ICER2 datalabel baseline = &base_ICER;
   yaxis discreteorder = data;
   xaxis grid display = (nolabel);
   refline 0 / axis = x;
   footnote j=right "Base case ICER = &base_ICER";
run; quit;
proc sgplot data = dtornado;
   label dICER1 = "discounted ICER with LOWER parameter value" dICER2 = "discounted ICER with HIGHER parameter value";
   hbar parameter / response = dICER1 datalabel baseline = &baseDICER;
   hbar parameter / response = dICER2 datalabel baseline = &baseDICER;
   yaxis discreteorder = data;
   xaxis grid display = (nolabel);
   refline 0 / axis = x;
   footnote j=right "Base case ICER = &baseDICER";
```

```
run; quit;
footnote;
```